

SVEUČILIŠTE U ZAGREBU
FILOZOFSKI FAKULTET
ODSJEK ZA INFORMACIJSKE I KOMUNIKACIJSKE ZNANOSTI
Ak. god. 2017./ 2018.

Josip Bartolović

Informacijski sustav u web okruženju:
interaktivna karta otvorenih profesorskih pozicija u školama u Hrvatskoj

Završni rad

Mentor: dr.sc. Kristina Kocijan, doc.

Zagreb 2018.

Sadržaj

1. Uvod.....	3
2. Osnovni pojmovi.....	5
2.1. Informacijski sustav	5
2.2. Baze podataka	5
2.3. Web 2.0	6
2.4. HTML.....	7
2.5. CSS.....	8
2.6. JavaScript.....	8
2.6.1. JSON.....	9
2.6.2. jQuery	10
2.6.3. API.....	10
3. Izrada projekta	12
3.1. Struktura podataka.....	12
3.2. HTML struktura web sjedišta.....	13
3.3. CSS struktura web sjedišta.....	15
3.4. Kreiranje Google Maps JavaScript karte	17
4. Zaključak.....	24
5. Literatura.....	25

Sažetak

U radu je opisan proces kreiranja informacijskoga sustava u obliku web-sjedišta. Sadržaj rada obuhvaća faze razvoja web-sjedišta; od početne ideje projekta do njegove svrhe, planiranje izrade web-sjedišta, informacijske sheme i strukture, kodiranje stranice u HTML i CSS kodu, izradu interaktivne karte uz pomoć Google Maps API-ja, programiranje u klijentskim i poslužiteljskim skriptnim jezicima, modeliranje baze podataka.

Ključne riječi: *informacijski sustav, interaktivna karta, HTML, CSS, JavaScript, Google Maps API*

Information system in a network environment : interactive map of opened professor positions in Croatia

Abstract

This paper describes process of creating an information system in form of web sites. The content of paper includes every phase of development; from the initial idea of project to its purpose, the planning of making web sites, information schemes and structures, coding page in HTML and CSS code, making interactive maps using Google Maps API, its programming in client and server scripting languages, and database modeling.

Key words: *information system, interactive map, HTML, CSS, JavaScript, Google Maps API*

1. Uvod

U ovom ću radu pisati o postupku izrade informacijskoga sustava u mrežnom okruženju: interaktivna karta koja prikazuje buduće otvorene profesorske pozicije u srednjim i osnovnim školama. Svrha je ovoga informacijskog sustava pomoći sadašnjim i budućim profesorima u pronalaženju radnoga mjesta.

Ovim informacijskim sustavom htio bih primarno olakšati odabir fakulteta i smjera studentima koji nisu sigurni u kojem smjeru žele nastaviti svoje obrazovanje, a kojima je primarni interes rad u školi kao profesor te olakšati profesorima potragu za radnim mjestom u školi. Sam sustav koristi generiranje interaktivne Google karte koja, prema odabranim parametrima iz izbornika, prikazuje gradove, odnosno njihove škole koje će za određeno radno mjesto tražiti nastavnike za unaprijed određene predmete.

Prednost je takvoga sustava što dobiveni podatci iz istoga služe kao podloga za prikaz budućnosti tržišta rada za poslove koje karta prikazuje te tako olakšavaju studentima izbor fakulteta, odnosno daju im okvirni pravac koji pokazuje u kojem smjeru bi se moglo kretati tržište rada. Mana je takvoga informacijskog sustava što zbog određene unutarnje i vanjske okoline izvori podataka koji se prikazuju na navedenoj interaktivnoj karti nisu u potpunosti točni. Tržište je rada turbulentno, te kao takvo podložno svakodnevnim izmjenama. Neke informacije poslova koje pruža informacijski sustav ovoga projekta, poput sigurnoga, otvorenoga radnog mjesta za određeni nastavni predmet u određenoj godini, svakako ne mogu biti u potpunosti točne. Primjerice, ako odredimo da će se u 2024. godini otvoriti radno mjesto za profesora informatike, takvi podatci mogu, ali i ne moraju biti točni zbog utjecaja unutarnje i vanjske okoline na trenutno zaposlenu osobu toga radnog mjesta te na samu instituciju koja takvo radno mjesto nudi. Zbog navedenih razloga takav sustav ne može biti automatski, te moraju postojati dodatni ljudski resursi koji će povremeno ažurirati podatke unutar baze podataka sukladno naputcima institucija i drugih tijela koje koriste isti sustav za posluživanje informacija.

U nastavku rada objasnit ću programske jezike korištene u izradi projekta te usporediti korištene tehnologije s drugim tehnologijama koje su mogle biti korištene za izradu ovoga projekta. Treći i četvrti dio rada posvećeni su postupku prikupljanja i strukturiranju potrebnih podataka te samom postupku izrade web-sjedišta. U zaključku rada ukratko ću navesti o čemu sam pisao te navesti plan za razvijanje ovoga informacijskog sustava u budućnosti.

2. Osnovni pojmovi

U ovom ću poglavlju navesti tehnologije koje su korištene u svrhu kreiranja ovoga projekta. Također ću objasniti funkcionalnost tehnologija korištenih u svrhu kreiranja projekta, njihovu utjecajnost u svijetu u usporedbi s drugim tehnologijama koje bi se mogle koristiti u izradi ovoga projekta te potkrijepiti primjerima u projektu.

2.1. Informacijski sustav

Prema Radovanu (1991) informacijski sustav prikuplja, pohranjuje, čuva, obrađuje i isporučuje potrebne informacije tako da su one dostupne svim članovima neke organizacije. U ovom projektu informacijski sustav služi za pohranjivanje podataka o školama, gradovima i županijama u Republici Hrvatskoj, obrađuje te isporučuje iste u obliku interaktivne zemljopisne karte nekoj organizaciji, odnosno ciljanim skupinama društva.

2.2. Baze podataka

Baza podataka je organizirani skup informacija ili podataka u računalno čitljivom obliku. Općenito je pristup takvim bazama moguć putem različitih online servisa za pretraživanje koje nazivamo i online sustavima odnosno informacijskim sustavima. Takvi servisi za pretraživanje imaju računala i programe (engl. *software*) koji omogućuju vanjskim korisnicima da pretražuju jednu ili više baza podataka kako bi mogli pronaći informaciju ili podatak gdje se ta informacija nalazi.

Postoje tri modela baza podataka - hijerarhijski, mrežni i relacijski. Svaki od ta tri modela na različit način organizira i upotrebljava podatke. Hijerarhijske i mrežne baze najčešće se koriste na velikim računalnim sustavima i na mini-računalima. Relacijska baza podataka organizira podatke u obliku tablice koja se sastoji od redaka i stupaca (Tuđman, Boras, & Dovedan, 1991).

2.3. Web 2.0

Web 2.0 je naziv koji se koristi za opisivanje druge generacije World Wide Weba, gdje statičke HTML stranice postaju interaktivne i dinamičke. Web 2.0 je usredotočen na ljudsku suradnju i dijeljenje informacija putem društvenih mreža, blogova i web-zajednica. Web 2.0 označava promjenu u kojoj je World Wide Web postao interaktivan doživljaj između korisnika i web izdavača, a ne jednosmjernog razgovora koji je prethodno postojao. Također predstavlja populističku verziju weba, gdje su novi alati omogućili da gotovo svatko može pridonijeti bez obzira na njihovo tehničko znanje („Web 2.0“, 2018).

Pojam „Web 2.0“ nastao je nakon O'Reilly Media konferencije 2004. godine nakon dot-com buma (engl. *dot-com bubble*). Iako naziv Web 2.0 sugerira da se radi o novoj verziji weba, on se ne odnosi na unaprijeđenje tehničkih specifikacija World Wide Weba, već na promjene u načinu na koji razvojni inženjeri koriste web-platformu. Web 2.0. ima nekoliko značajnijih karakteristika poput omogućenog korištenja aplikacije u potpunosti kroz web preglednik, kontrola korisnika nad sadržajem stranice kroz različite razine pristupa, unaprijeđena grafička sučelja u odnosu na Web 1.0 (CARNet, 2009). Neke Web 2.0 tehnologije uključuju:

Blog - blogovi su se pojavili sredinom devedesetih godina te ih s pravom možemo smatrati najstarijima među Web 2.0 aplikacijama. Iako su prvotno zamišljeni kao online dnevnici, s vremenom su postali opći alati informiranja

Wiki - repozitoriji znanja u obliku web stranice na kojoj korisnici mogu dodati, obrisati ili urediti sav sadržaj, ponekad i bez potrebe da se registriraju. Najpoznatiji primjer wiki stranice je Wikipedija.

Društvene knjižne oznake - Društvene knjižne oznake (engl. *social bookmarking*) su način na koji zaposlenici mogu pohranjivati, klasificirati, pretraživati i dijeliti svoje interese online.

Podcasting - Osnovna namjena podcastinga je stvaranje vlastitih multimedijalnih datoteka koje se mogu vrlo jednostavno publicirati na Webu.

Društvene mrežne stranice (engl. *Social Network Sites*) - Web servisi u obliku virtualnih zajednica gdje se “susreću” ljudi istih interesa. Pojedinci stvaraju profile s osobnim podacima, znanjima, interesima i vještinama te pomoću njih ostvaruju kontakte s istomišljenicima. (Orehovački, Konecki, & Stapić, 2008)

2.4. HTML

Osnovni jezik koji upotrebljavamo za izradu internetskih stranica je HTML¹. On našim internetskim preglednicima daje podatke o sadržaju i strukturi učitane web stranice, a preglednik od tih podataka oblikuje stranicu kakvu mi vidimo. Definirao ga je 1990. godine Timothy Berners-Lee, danas ravnatelj W3C konzorcija, organizacije koja brine o standardizaciji web tehnologija i razvoju weba („Uvod u HTML“, n.d.).

U izradi ovoga web-sjedišta korištena je HTML verzija HTML5. Kako bismo odredili koju verziju HTML-a koristimo, pišemo deklaraciju `<!DOCTYPE html>` na početku dokumenta prije prve `<html>` oznake. Dodavanjem ove deklaracije preglednik će znati da koristimo HTML5 verziju dokumenta. Kako bi stranica ispravno funkcionirala, odnosno kako bi se dijakritički znakovi u hrvatskome jeziku ispravno prikazali, potrebno je odrediti koji ćemo set znakova koristiti pomoću `<meta>` oznake. U ovom projektu korišten je UTF-8 set. Prema tome, koristit ćemo oznaku `<meta charset= "UTF-8">`. Primjer jednostavnoga HTML dokumenta prikazan je na slici 1.

¹ Kratica za HyperText Markup Language


```
<!DOCTYPE html>
<html>
  <head>
    <meta charset = „utf-8“>
    <title>Naziv Stranice</title>
  </head>
  <body>
    <p>Sadržaj stranice</p>
  </body>
</html>
```

Slika 1: Primjer HTML dokumenta

2.5. CSS

Cascading Style Sheets (CSS) stilski je jezik koji opisuje kako će HTML elementi stranice biti prikazani u pregledniku poput fonta, boje, veličine ili rasporeda sadržaja prikazanoga na web-stranici („CSS“, 2017).

Mujadžević (2014) navodi, CSS-pravilo može se napisati tako da bude primijenjivo na sve elemente kao i na određenu skupinu elemenata ili zasebne elemente. Prije pojave CSS-a problem u oblikovanju izgleda web-stranice bio je u tome da je vizualni prikaz moguće postići i u HTML-u što je u konačnici kod činilo nepreglednim. Pojavom CSS-a uvelike se olakšava posao web dizajnera jer je ideja CSS-a bila odvajanje prezentacijskog kôda u zasebnu datoteku. Prva verzija CSS-a definirana je krajem 1996.

2.6. JavaScript

Kada se JavaScript prvi put pojavio 1995. godine, njegova je glavna svrha bila obraditi dio validacije ulaznih podataka koja je prethodno bila prepuštena jezicima na poslužitelju. Prije toga, putovanje ulaznih podataka na poslužitelj bilo je potrebno kako bi se utvrdilo je li obavezno polje bilo prazno ili unesena vrijednost nije valjana.

Netscape Navigator je nastojao to promijeniti uvođenjem JavaScript-a. Mogućnost upravljanja nekim osnovnim provjerama klijenta bila je uzbudljiva nova značajka u isto vrijeme kada je uporaba telefonskih modema bila raširena. Od tada, JavaScript je postao važna značajka svakog većeg web preglednika. Više se ne obvezuje

na jednostavnu provjeru podataka te sada komunicira sa skoro svim aspektima prozora preglednika i njegovog sadržaja (Zakas, 2005).

Drugi jezik, koji je mogao biti korišten umjesto JavaScript jezika, je PHP. Khillar (2018) opisuje JavaScript kao je posebno dizajniran jezik za izgradnju interaktivnih web-sučelja i stvaranje aplikacija specifičnih za mrežu bez interakcije s poslužiteljem dok je PHP jezik zadužen za sve servisne funkcije poput izgradnje prilagođenog web-sadržaja, autentifikacije korisnika, rukovanja zahtjevima te prikazivanja rezultata.

Sam JavaScript jezik nema mogućnosti interakcije sa serverom, no kako JavaScript svakodnevno napreduje mnogo brže naspram PHP jezika, pojavljuje se Node.js kao poslužiteljsko okruženje u svrhu servera i serverskih modula koji služe isključivo za interakciju s JavaScript jezikom. Samim time JavaScript, kao klijentsko poslužiteljski jezik, napreduje do, ili čak iznad, serverskih mogućnosti PHP jezika te je, zbog svrhe samoga jezika i njegovoga globalnog napretka, JavaScript izabran u svrhu ovoga projekta kao primarni skriptni jezik.

2.6.1. JSON

JavaScript Object Notation (JSON) tekstualni je otvoreni standard pisan u JavaScript sintaksi. Služi za pohranjivanje i razmjenu podataka (W3Schools, 2018).

Drugi je format podataka koji se mogao koristiti u svrhu izrade ovoga projekta XML. Svrha je XML formata podataka označavanje dokumenta putem tagova (slično HTML-u), što nije svrha JSON formata. Prednost je XML-a što različiti podatci mogu biti strukturirani kroz više istoimenih tagova, dok unutar JSON formata svako svojstvo mora biti jedinstvenoga naziva. Samim time JSON ima prednost nad XML formatom zbog smanjene strukture, bolje i lakše iskoristivosti već klijentu proslijeđenih podataka, lakšega i jednostavnijega koda za pisanje te zahtijeva manje trošenje računalnih resursa u odnosu na XML.

Najvažnije su prednosti JSON-a u odnosu na XML što se JSON parsira kroz standardnu JavaScript funkciju, dok se XML parsira kroz XML parser („JSON“, 2018) te je JSON podskup JavaScript jezika, tako da struktura koda više odgovara samom jeziku. Zbog navedenih prednosti JSON formata u sklopu JavaScript jezika, korišten je u ovom projektu kao struktura i sintaksa baze podataka.

2.6.2. jQuery

jQuery je posebna vrsta JavaScript biblioteke, razvijene s namjenom da bude nadogradnja osnovnoga JavaScripta. jQuery pojednostavljuje njegovu sintaksu i omogućava lakšu uporabu te bolju interakciju između JavaScripta i drugih programskih jezika namijenjenih razvoju mrežnih aplikacija. Osnovna je ideja jQuery-ja „Piši manje, učini više!“ (W3Schools, 2018).

U ovom je projektu jQuery korišten za olakšano pozivanje JSON datoteke naspram standardnoga JavaScript načina, te kako bi pomoću iste funkcije dohvatili JSON objekt i koristili podatke iz baze za potrebe aplikacije.

2.6.3. API

API je akronim za aplikacijsko programsko sučelje (engl. *Application Programming Interface*), tj. softverski posrednik koji omogućuje dvjema aplikacijama da međusobno komuniciraju. Kada klijent upotrebljava aplikaciju, program se spaja na internet i šalje podatke poslužitelju. Poslužitelj zatim preuzima te podatke, interpretira ih, obavlja potrebne radnje i šalje ga natrag klijentu. Aplikacija zatim tumači te podatke i prikazuje vam informacije koje ste htjeli na čitljiv način („What is an API?“, 2018).

API navodi mnoštvo operacija koje razvojni programeri mogu koristiti, zajedno s opisom onoga što rade. Razvojni programer ne mora nužno znati kako se, na primjer, gradi operativni sustav. Oni samo trebaju znati da su operacije dostupne za upotrebu u svojoj aplikaciji. Također, razvojni programeri mogu dostaviti vlastite podatke API-ju kako bi dobili rezultate. API često igra veliku ulogu u sigurnosti. Koristi se i za kontrolu

pristupa hardverskim uređajima i softverskim funkcijama za koje aplikacija možda ne mora nužno imati dozvolu za upotrebu (Hoffman, 2018).

U radu je korišten Google Maps API. Kroz njegove funkcije, te funkcije same aplikacije, omogućena je komunikacija između servera na kojemu se nalaze podaci i klijenta. On nam omogućuje posluživanje interaktivne zemljopisne karte s podacima koje klijent pretražuje, a koji se nalaze unutar baze podataka na serveru.

3. Izrada projekta

3.1. Struktura podataka

Prvi je korak u izradi samoga web-sjedišta prikupljanje podataka koji će biti prikazani na interaktivnoj karti. Podatci o školama ručno su dohvaćani sa stranice e-Usmjeravanje. Uz podatke o školama potrebne su nam koordinate svih mjesta u kojima se te škole nalaze kao i koordinate županija kako ne bi bilo potrebno ručno pretraživati kartu. Koordinate su također ručno dohvaćene s Google maps stranice. Prikupljeni podatci su strukturirani u JSON objekt. Predložak je prikazan na slici 2.

```
{
  "Bjelovarsko-bilogorska" : {
    "long": "45.7354556",
    "lng" : "16.8454265",
    "cities":
      [
        {
          "city" : "Bjelovar",
          "lat": "45.9062305",
          "lng" : "16.7941253",
          "info" : {
            "2018" : {
              "skola" : "Medicinska škola",
              "predmet" : "Farmakologija",
              "kontakt" : "med-skola-bjelovar@bj.t-com.hr"
            },
            "2019" : {
              "skola" : "Medicinska škola",
              "predmet" : "Patologija i patofiziologija",
              "kontakt" : "med-skola-bjelovar@bj.t-com.hr"
            },
            "2020" : {
              "skola" : "Medicinska škola",
              "predmet" : "Higijena i preventivna medicina",
              "kontakt" : "med-skola-bjelovar@bj.t-com.hr"
            },
            "2021" : {
              "skola" : "Medicinska škola",
              "predmet" : "Profesionalna komunikacija u sestrinstvu",
              "kontakt" : "med-skola-bjelovar@bj.t-com.hr"
            },
            "2022" : {
              "skola" : "Medicinska škola",
              "predmet" : "Zdravstvena njega u zaštiti mentalnog zdravlja",
              "kontakt" : "med-skola-bjelovar@bj.t-com.hr"
            }
          }
        }
      ]
  }
},
```

Slika 2: predložak JSON objekta

3.2. HTML struktura web sjedišta

U poglavlju 2.4 naveden je primjer jednostavnog HTML dokumenta. Na tom primjeru postupno ću dodavati sadržaj stranice.

Element `<head>` sadrži elemente koji opisuju sam html dokument ili ga povezuju s vanjskom datotekom kao što su naslov stranice, CSS ili JavaScript datoteka što ću spomenuti u daljnjem dijelu rada. `<title>` element pojavljuje se na naslovnoj traci web-preglednika gdje je prikazan naziv stranice. Kako je vidljivo na slici 3, naziv je ove



Slika 3: Naziv stranice

stranice Projekt.

Unutar `<body>` elementa nalazi se sadržaj koji se prikazuje na web-stranici kao i zaglavlje (engl. *header*) i podnožje (engl. *footer*). U zaglavlju se nalazi navigacijska traka koja povezuje stranice web-sjedišta i naslov. Navigacijsku traku definirat ćemo `<nav>` elementom te ćemo koristiti elemente `` i `` koji služe za izradu neuređene liste te element `<a>` kojem ćemo definirati poveznice na ostale stranice. Kako bismo omogućili povezivanje stranica, elementu `<a>` dodat ćemo atribut `href`. Atributu `href` za vrijednost dodjeljujemo imena HTML dokumenata. Struktura navigacijske trake prikazana je na slici 4.

```
<nav>
  <ul class="nav">
    <li><a href="index.html">Naslovnica</a></li>
    <li><a href="contact.html">Kontakt</a></li>
    <li><a href="map.html">Karta</a></li>
  </ul>
</nav>
```

Slika 4: Navigacijska traka

Podnožje web-sjedišta definirano je elementom `<footer>`. Podnožje u ovom projektu nema važnoga sadržaja. Unutar njega nalazi se slika i naziv sveučilišta, fakultet i odsjek. Struktura podnožja prikazana na slici 5, a završni izgled web-stranice na slici 6.

```
<footer>
    
    <div class="innerFooter">
        <div class="footerText">Sveučilište u Zagrebu | Filozofski fakultet |
        | Odsjek za informacijske i komunikacijske znanosti</div>
    </div>
</footer>
```

Slika 5: Struktura podnožja

```
Naslovnica  Kontakt  Karta
<section class="homeBody-map">

    <section id="map" class="mapWrapper"></section>

    <div class="form-style-2">
        <div class="form-style-2-heading">Pretraži gradove</div>

        <label for="field1"><span>Županija <span class="required">*</span></span> <select id="regionsMenu" name="years" class="input-field">
        </select></label>
        <label for="field2"><span>Godina <span class="required">*</span></span> <select id="y" name="years" class="input-field"></select></label>

        <label><span>&nbsp;</span></span><input type="submit" value="Pretraži" onclick="search()"/></label>
    </div>
</section>
```

Slika 7: HTML struktura izbornika

Sveučilište u Zagrebu | Filozofski fakultet | Odsjek za informacijske i komunikacijske znanosti

Stranica s kartom sastoji se od same karte i izbornika koji su potrebni kako bismo mogli prikazati podatke na karti. Pomoću `<select>` elementa definirana su dva padajuća izbornika. Prvi izbornik služi za odabir željene županije, a drugi za odabir godine. Odabir županije preuzima se iz JSON datoteke, stoga ih nije potrebno ručno definirati pomoću elementa `<option>`. Kako bismo uopće mogli pretražiti podatke, potrebno je

Slika 6: Završni izgled naslovne stranice

dodati gumb za slanje zahtjeva pomoću `<input>` oznake kojoj je dodana vrijednost `submit`. Na slici 7 prikazana je struktura izbornika. Oznaka `<section id="map" class="mapWrapper"></section>` namijenjena je za prikaz i pozicioniranje karte. Nešto više o tome ću pisati u daljnjem dijelu rada.

3.3. CSS struktura web sjedišta

U ovom projektu korišten je vanjski CSS stilski list. Korištenje vanjske CSS datoteke uvelike olakšava posao te HTML kod čini preglednijim u urednijim. Jednim stilskim listom možemo upravljati izgledom svih stranica web-sjedišta. CSS datoteka u `<head>` elementu mora imati poveznicu pomoću elementa `<link>` kako je prikazano na

```
<head>
<link rel="stylesheet" href="css/style.css" />
</head>
```

Slika 8: Poveznica vanjske CSS datoteke

slici .

U ovom CSS dokumentu opisani su elementi zajednički svim stranicama kao i elementi koji pripadaju pojedinim stranicama. Navigacijska je traka opisana nav

```
nav {
    position: relative;
}
.nav {
    display: flex;
    align-items: center;
    position: absolute;
    height: 100%;
    width: 100%;
    border-width: 1px 0;
    list-style: none;
    margin: 0;
    padding: 0;
    text-align: center;
    background-color: rgb(36, 49, 66);
}
.nav li {
    display: flex;
    align-items: center;

    padding: 0 2% 0 2%;
    height: 100%;
    transition: 0.2s;
}
.nav a {
    display: inline-block;
    color: white;
    text-decoration: none;
    font-size: 40px;
    transition: 0.2s;
}
```


selektorom. Sama CSS struktura navigacijske trake prikazana na slici 9.

Svojstvo `display` određuje način kako je element prikazan na zaslonu. Pomoću vrijednosti `inline-block` poveznice u navigacijskoj traci postaju blok elementi u horizontalnom položaju. Vrijednost `flex` proširuje elemente kako bi popunili slobodni prostor ili smanjuje kako bi se spriječilo „prelijevanje“ elemenata.

Slika 9: Oblikovanje navigacijske trake u CSS datoteci

3.4. Kreiranje Google Maps JavaScript karte

Nakon modeliranja baze podataka u JSON formatu potrebno je odrediti funkcionalnost sučelja karte, odnosno njezinoga izbornika. Funkcionalnosti koje su potrebne za ovaj rad jesu:

- generiranje karte prema unaprijed zadanim parametrima pri učitavanju dokumenta
- generiranje izbornika za odabir županija
- generiranje izbornika za odabir godina
- generiranje karte prema odabranim parametrima putem prethodno navedenih izbornika:
 - generiranje info-markera za odabrano područje prikazano na karti
 - lebdeći (Hover) efekt za prethodno navedene markere za prikaz informacija koje sadrže.

Prvi korak, prije izrade ranije navedenih funkcionalnosti, potrebno je kreirati HTML elemente unutar kojih će se one nalaziti. U sklopu ovoga projekta i korištenih tehnologija, isti HTML elementi mogu biti kreirani putem HTML jezika ili JavaScript jezika. U ovome radu elementi su kreirani putem HTML jezika te je istima dodijeljen atribut id sa zadanom vrijednosti. Za funkcionalnost karte potrebni su sljedeći elementi:

- element koji sadržava samu kartu
 - `<section id="map" class="mapWrapper"> </section>`
- element koji će sadržavati izbornik županija
 - `<select id="regionsMenu" class="input-field"> </select>`
- element koji će sadržavati izbornik godina

○ `<select id="year" class="input-field">`²

Nakon kreiranja prethodno navedenih HTML elemenata, potrebno ih je putem zadanoga id atributa dohvatiti kroz JavaScript jezik kako bismo mogli unutar njih

```
const mapContainer = document.getElementById('map');
const regionsContainer = document.getElementById("regionsMenu");
const yearsContainer = document.getElementById("year");
const defaultMapCoords = {lat:45.4745666, lng:16.3557017};
```

Slika 10: Konstante za generiranje mape i povezivanje izbornika

generirati ostale elemente, odnosno unijeti ostale potrebne podatke prikazane na slici 10.

Konstanta **mapContainer** dohvaća id HTML elementa `<div>` namijenjenoga za prikaz Google mape. Konstanta **regionsContainer** dohvaća id HTML elementa `<select>` namijenjenoga za pohranu županija. Konstanta **yearsContainer** dohvaća id HTML elementa `<select>` namijenjenoga za pohranu godina, a **defaultMapCoords** konstanta sadrži koordinate početne mape prilikom učitavanja stranice.

Sljedeće je potrebno definirati poziv na samu bazu podataka prikazanoga na slici 11. U ovom radu korištena je AJAX (asynchronous JavaScript and XML) tehnologija iz jQuery biblioteke. Razlog korištenja jQuery biblioteke za rad s AJAX tehnologijom naspram običnoga JavaScript pristupa olakšani je pristup i korištenje istih, odnosno

```
function callJSON(url, callback) {
  $.ajax({
    method: "GET",
    url: url,
    dataType: "json",
  })
  .done(function(request){
    callback(request);
  })
}
```

Slika 11: Kreiranje funkcije poziva

² Elementi su prethodno prikazani unutar kodnoga boka 4

sličnih poziva na datoteke istoga ili drugoga tipa te dohvaćanje podataka.

Nakon kreiranja poziva prema bazi podataka definiramo funkcije koje će unutar prethodno navedenih `<select>` HTML elemenata kreirati `<option>` HTML elemente s izvezenim podacima iz baze podataka. Radi jednostavnije kontrole kreiranja podataka te strukture koda, dobivanje podataka županija iz baze podataka razdvojeno je dvjema funkcijama³ (slika 12 i slika 13). U prvoj funkciji generiramo samo jedan `<option>`

```
function regionsMenu (region, container) {  
    let yOpt = document.createElement("option");  
    yOpt.value = region;  
    yOpt.innerHTML = region;  
    container.appendChild(yOpt);  
}
```

Slika 12: Funkcija za generiranje `<options>` elementa za županije

HTML element:

Nakon toga navedenu funkciju unosimo unutar druge funkcije, kroz for petlju, kako bismo generirali više prethodno navedenih elemenata, no sa zasebnim podacima:

Atribut `data` predstavlja JSON objekt baze podataka koji će putem *callback* funkcije iz AJAX poziva, odnosno njegove roditeljske funkcije, biti proslijeđen u navedenu funkciju. Od tud će se izvući podatci županija te umetnuti u `<option>` HTML

```
function generateRegionsMenu(data) {  
    for(var i = 0; i <= Object.keys(data).length-1; i++) {  
        regionsMenu(Object.keys(data)[i], regionsContainer);  
    }  
}
```

Slika 13: Funkcija za generiranje cjelovitog izbornika županija

```
callJSON("javascript/maps-data.json", function(data){  
    generateRegionsMenu(data);  
})
```

Slika 14: Izbornik županija

elemente kako bi u konačnici dobili izbornik za odabir županija (slika 14).

Kada su izrađene prethodno navedene funkcionalnosti, tada dolazi do izrade funkcionalnosti generiranja karte prema odabiru vrijednosti obaju izbornika koji su navedeni prethodno u tekstu. Dolazi do kreiranja funkcije **fetchData(data)** s jednim zadanim argumentom *data* koji predstavlja JSON objekt baze podataka, no s obzirom na njezinu veličinu, u daljnjem tekstu bit će objašnjena kroz njezine manje dijelove.

Unutar funkcije **fetchData(data)** prvo definiramo varijable koje će samo ta funkcija koristiti, a koje su potrebne za funkcionalnost generiranja karte:

- varijabla za info prozor gradova prikazanih na generiranoj karti
 - `let infoWindow = new google.maps.InfoWindow();`
- dohvaćanje selektiranoga podatka izbornika županija
 - `region=regionsContainer.options[regionsContainer.selectedIndex].value;`
- dohvaćanje selektiranoga podatka izbornika godina
 - `year = yearsContainer.options[yearsContainer.selectedIndex].value;`
- dohvaćanje grada iz baze podataka na temelju prethodnih parametara
 - `let cities = data[region].cities;`
- `let markets = [];` -> Pomoćno polje za generiranje markera na karti

Nadalje, potrebno je odrediti objekt koji će služiti kao predložak za generiranje karte

```
let map = new google.maps.Map(document.getElementById('map'), {  
  zoom: 8.5,  
  center: new google.maps.LatLng(data[region].long, data[region].lng),  
  content: cities,  
  mapTypeId: google.maps.MapTypeId.ROADMAP  
});
```

Slika 15: Predložak za generiranje karte

prema zadanim, nekim prethodno navedenim parametrima i svojstvima (slika 15).

Pomoću for petlje slici 16 te prethodnih parametara generiramo kartu prema odabranoj županiji, markere za gradove unutar te županije te info prozore navedenih

```
for(var i = 0; i <= cities.length; i++) {  
    let coords = {lat: cities[i].lat, lng: cities[i].lng};  
    let contentString =  
        '<div id="">' +  
        '<div id="">' +  
        '</div>' +  
        '<h1>' + cities[i].city + '</h1>' +  
        '<div>Škola: ' + cities[i].info[year].skola + '</div>' +  
        '<div>Predmet: ' + cities[i].info[year].predmet + '</div>' +  
        '<div>Kontakt: ' + cities[i].info[year].kontakt + '</div>' +  
        '</div>';  
  
    markers[i] = new google.maps.Marker({  
        position: coords,  
        map: map,  
        content: contentString,  
        id: i  
    });  
    google.maps.event.addListener(markers[i], 'mouseover', function () {  
        infoWindow.setContent(markers[this.id].content);  
        infoWindow.open(map, this);  
    })  
    google.maps.event.addListener(markers[i], 'mouseout', function () {  
        infoWindow.close(map, this);  
    })  
}
```

Slika 16: Generiranje markera karte i njihovih funkcionalnosti

markera s njihovim podacima.

Petlja sadrži sljedeće:

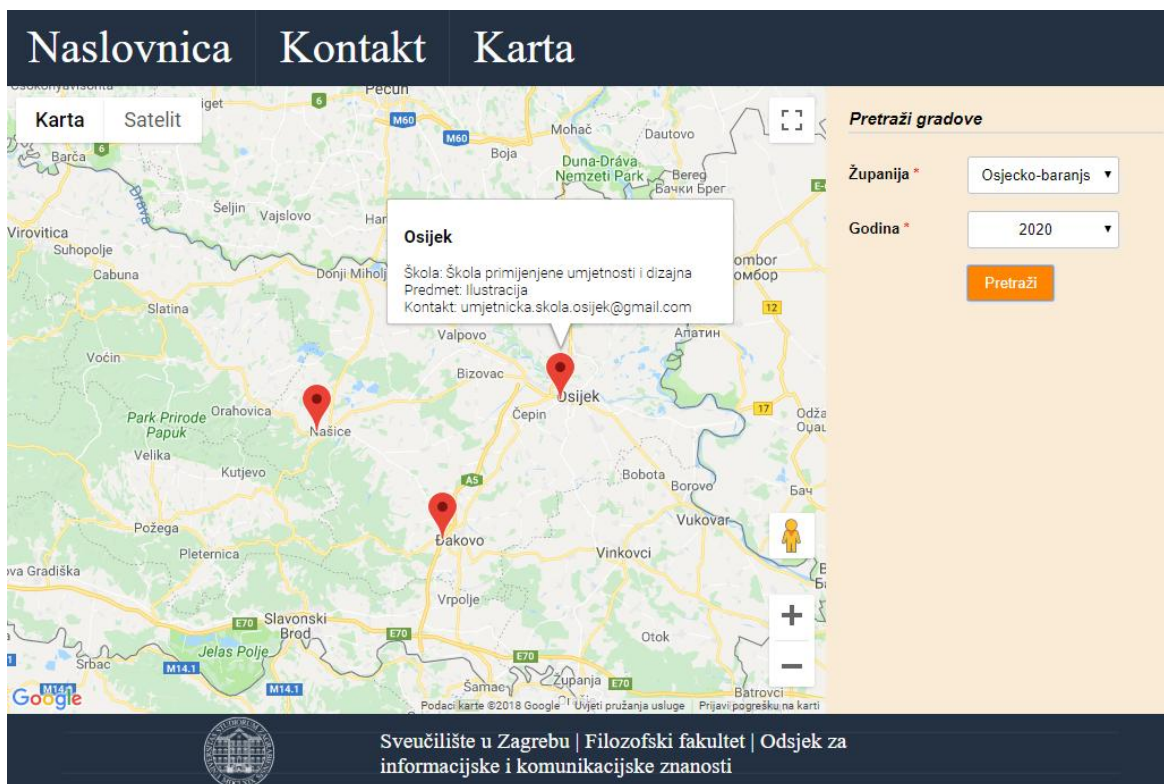
- početak petlje sa zadanim parametrima

- varijablu `contentString` koja sadrži podatke info prozora svakoga markera te se koristi kao njegov predložak
- pomoćnim poljem podataka `markers` te Google Maps API generiramo markere za gradove prema navedenim svojstvima objekta:
 - `Position` – koordinate grada
 - `Map` – prethodno definirana mapa unutar iste funkcije sa zadanim vrijednostima
 - `Content` – Predložak info prozora s proslijeđenim podacima samoga grada
 - `id` - njegov id koji će se koristiti naknadno unutar koda
- pomoću dodavanja JavaScript eventa, odnosno putem predefiniranoga Google Maps API objekta i funkcijom dodajemo “`mouseover`” event koji prilikom prelaska mišem preko markera otvara info prozor s podacima izvezenim iz baze podataka
- za zatvaranje prethodno navedenoga info prozora, na isti način dodajemo “`mouseout`” event.

Kada smo u potpunosti kreirali funkcionalnost funkcije dobivanja podataka iz same baze, potrebno je istu pozvati putem klika na gumb koji se nalazi u sklopu izbornika

```
function search() {
  callJSON("javascript/maps-data.json", function(data){
    fetchData(data);
  })
}
```

Slika 17: Funkcija koja dohvaća podatke za prikaz na interaktivnoj karti županija i godina. Poziv funkcije odrađuje tako da istu prosljedimo prethodno navedenoj funkciji callJSON koja prije svega izvršava poziv na bazu podataka te dohvaća objekt iz iste kako bi se mogao upotrijebiti za daljnji rad aplikacije (slika 17). Završni izgled karte prikazan je na slici 18.



Slika 18: Završni izgled karte

4. Zaključak

Ideja projekta bila je izraditi interaktivni informacijski sustav koji će na interaktivnoj karti prikazati dostupna radna mjesta profesora u srednjim školama u razdoblju od 2018. do 2022. godine u željenoj županiji. Korisnici mogu vidjeti podatke o školi kao što su naziv škole, predmet za koji je otvoreno radno mjesto i kontakt škole.

Projekt trenutno sadrži informacije samo o srednjim školama te ne postoji mogućnost odabira određenoga predmeta. S obzirom na to da je ovaj projekt u testnoj fazi, u budućnosti planiram dodati podatke i za osnovne škole te omogućiti korisnicima više mogućnosti u izborniku.

Činjenica je da nismo svi sigurni koje zanimanje odabrati i baviti se time većinu našega života. Ova odluka za pojedinca može biti iznimno stresna. Iz toga razloga svrha je ovoga projekta pomoći budućim studentima koji se ne mogu opredijeliti za studij, a žele steći profesorsko zanimanje, pri odabiru željenoga fakulteta. Također, ovaj projekt nije namijenjen isključivo budućim studentima, već i profesorima kako bi im olakšao u pronalasku radnoga mjesta, te ostalim osobama koje žele postati profesori u određenome polju predmeta.

Kako je već u uvodnome dijelu rada navedeno, podatci prikupljeni putem takvoga sustava mogu okvirno prikazati budućnost toga dijela tržišta rada kako bi, uz mogućnost pronalaska posla, prikaza natječaja za otvorena radna mjesta, osobe koje koriste ovakav sustav mogle uvidjeti kolika će biti potražnja za njima interesantnim radnim mjestima. Iako učenicima i studentima takvi podatci olakšavaju izbor odabira fakulteta, tržište je rada promjenjivo i turbulentno te isti podatci ne mogu biti potpuno točni za određeni vremenski period. Glavni su razlozi tome unutarnji i vanjski čimbenici okoline ustanova koje koriste takav sustav te putem njega prikazuju trenutno stanje posluženih podataka. Iznimno je potrebno da klijenti prate informacije takvoga interaktivnog sustava u češćim vremenskim periodima te poslužitelji dodatnim ljudskim resursima održavaju točnost podataka.

5. Literatura

1. „CSS”. (2017). preuzeto s <https://developer.mozilla.org/en-US/docs/Glossary/CSS> (posjećeno 15.8. 2018.),
2. „JSON (format za razmenu podataka)“. (2018), preuzeto s <https://www.webprogramiranje.org/json/>, (posjećeno 19.8.2018)
3. „Uvod u HTML“. (n.d.). preuzeto s: <https://tesla.carnet.hr/mod/book/view.php?id=5430&chapterid=885> (posjećeno: 14.9.2018)
4. „What is an API? (Application Programming Interface)“. (2018). preuzeto s: <https://www.mulesoft.com/resources/api/what-is-an-api> (posjećeno 14.9.2018)
5. CARNet. (2009). Društveni softver: Web 2.0. preuzeto s <http://www.carnet.hr/tematski/drustvenisoftver/pojmovnik>, (posjećeno:12.8.2018.),
6. Hoffman, C. (2018). What is an API?. preuzeto s <https://www.howtogeek.com/343877/what-is-an-api/> (posjećeno 14.9.2018)
7. Khillar, S. (2018). Difference between JavaScript and PHP. preuzeto s <http://www.differencebetween.net/technology/difference-between-javascript-and-php/> (posjećeno 7.9.2018)
8. Mujadžević, E. (2014). Uvod u CSS. Zagreb, Hrvatska: Sveučilišni računski centar „Srce“
9. Radovan, M. (1991.). Projektiranje informacijskih sustava. Zagreb, Hrvatska: Informator
10. Tuđman M., Boras D., Dovedan Z. (1991). Odrednice informacijskih sustava. Uvod u informacijske znanosti. preuzeto s <http://www.ffzg.unizg.hr/infoz/dzs/text/Uvod%20u%20informacijske%20znanosti/> (posjećeno 6.8.2018.),
11. Tuđman, M., Boras, D., Dovedan, Z. (1991). Baze podataka. Uvod u informacijske znanosti. preuzeto s

- <http://www.ffzg.unizg.hr/infoz/dzs/text/Uvod%20u%20informacijske%20znanosti/> (posjećeno 6.8.2018.),
12. W3Schools.com. (2018). jQuery Tutorial. preuzeto s <http://www.w3schools.com> (posjećeno 16.8.2018)
13. W3Schools.com. (2018). JSON – Introduction. preuzeto s <http://www.w3schools.com>, (posjećeno 16.8.2018)
14. Zakas, N. C. (2005). Professional JavaScript for Web Developers. Indianapolis, Indiana: Wiley Publishing, Inc.
15. „Web 2.0“. (2018). preuzeto s <https://www.techopedia.com/definition/4922/web-20> (posjećeno 14.9.2018)